

[illegible]

```

LL          IIIIII          SSSSSSSS
LL          IIIIII          SSSSSSSS
LL          II             SS
LL          II             SS
LL          II             SS
LL          II             SS
LL          II             SSSSSS
LL          II             SSSSSS
LL          II             SS
LL          II             SS
LL          II             SS
LL          II             SS
LL          II             SSSSSS
LLLLLLLLLLLL IIIIIIII          SSSSSSSS
LLLLLLLLLLLL IIIIIIII          SSSSSSSS

```

SHODEF
Table of contents

- MONITOR SHOW DEFAULT Command H 13

16-SEP-1984 02:05:00 VAX/VMS Macro V04-00

Page 0

(2) 60
(4) 136
(14) 660
(20) 883

DECLARATIONS
SHODEF_CMD - MONITOR SHOW DEFAULT command
SHOW_CLASSES - Show all selected classes
SHOW_A_LINE - Put a line of SHOW to terminal

SHO
V04

63

63

63


```
0000 1      .TITLE SHODEF - MONITOR SHOW DEFAULT Command
0000 2      .IDENT 'V04-000'
0000 3
0000 4
0000 5      *****
0000 6      *
0000 7      *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8      *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9      *  ALL RIGHTS RESERVED.
0000 10     *
0000 11     *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12     *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13     *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14     *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15     *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16     *  TRANSFERRED.
0000 17     *
0000 18     *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19     *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20     *  CORPORATION.
0000 21     *
0000 22     *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23     *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24     *
0000 25     *
0000 26     *****
0000 27
0000 28     ++
0000 29     FACILITY: VAX/VMS MONITOR Utility
0000 30
0000 31     ABSTRACT:
0000 32
0000 33         The SHODEF module executes the SHOW DEFAULT subcommand
0000 34         of the MONITOR utility. It is called by the CLE (Command
0000 35         Language Editor).
0000 36
0000 37     ENVIRONMENT:
0000 38
0000 39         User mode, IPL 0, unprivileged.
0000 40
0000 41     AUTHOR: Thomas L. Cafarella, March, 1983
0000 42
0000 43     MODIFIED BY:
0000 44
0000 45         V03-003 PRS1016      Paul R. Senn      04-Apr-1984      14:00
0000 46         Use $PARSE to expand filespecs and hide passwords.
0000 47
0000 48         V03-003 PRS1013      Paul R. Senn      28-Mar-1984      14:00
0000 49         Give SHOW DEFAULT the ability to handle multiple input files.
0000 50
0000 51         V03-002 PRS1011      Paul R. Senn      29-Feb-1984      14:00
0000 52         add /FLUSH_INTERVAL qualifier
0000 53
0000 54         V03-001 PRS1001      Paul R. Senn      27-Dec-1983      16:00
0000 55         Make default interval = 6 for ALL classes Pseudo-class
0000 56         live requests.
0000 57
```


SHODEF
V04-000

- MONITOR SHOW DEFAULT Command

J 13

16-SEP-1984 02:05:00 VAX/VMS Macro V04-00
5-SEP-1984 02:02:35 [MONITOR.SRC]SHODEF.MAR;1

Page 2
(1)

0000 58 ;--

SHC
V04

```
0000 60      .SBTTL  DECLARATIONS
0000 61      .PSECT  MONDATA,QUAD,NOEXE
0000 62      :
0000 63      : INCLUDE FILES:
0000 64      :
0000 65      :
0000 66      $CDBDEF      ; Define Class Descriptor Block
0000 67      $MRBDEF      ; Define Monitor Request Block
0000 68      $MONDEF      ; Monitor Recording File Definitions
0000 69      $DSCDEF      ; Descriptor Definitions
0000 70      $IFBDEF      ; File descriptor table definitions
0000 71      :
0000 72      :
0000 73      : MACROS:
0000 74      :
0000 75      :
0000 76      :
0000 77      : Local Macro Definitions
0000 78      :
0000 79      :
0000 80      :
0000 81      : ALLOC Macro - Dynamically allocate space on the stack.
0000 82      :
0000 83      :
0000 84      .MACRO  ALLOC    LENGTH,RSLDESC,RSLBUF
0000 85      SUBL    #<LENGTH+3>&<^C3>,SP
0000 86      .IF     NB,RSLBUF
0000 87      MOVL    SP,RSLBUF
0000 88      .ENDC
0000 89      PUSHL    SP
0000 90      PUSHL    #LENGTH
0000 91      MOVL    SP,RSLDESC
0000 92      .ENDM    ALLOC
0000 93      :
0000 94      :
0000 95      : EQUATED SYMBOLS:
0000 96      :
0000 97      :
```



```
0000 99 ;
0000 100 ; OWN STORAGE
0000 101 ;
0000 102 ;
61 76 20 64 65 64 72 6F 63 65 72 00' 0000 103 RV_STR: .ASCIC \recorded value\ ; Text for playback values
65 75 6C 000C
0E 0000
6D 69 74 20 74 6E 65 72 72 75 63 00' 000F 104 CT_STR: .ASCIC \current time\ ; Text for /BEGINNING
65 001B
0C 000F
65 74 69 6E 69 66 65 64 6E 69 00' 001C 105 ID_STR: .ASCIC \indefinite\ ; Text for /ENDING
0A 001C
41 21 3C 39 21 2F 0000002F'010E0000' 0027 106 CS_SEG1: .ASCID \!/9<!AS!> = !27<!\ ; Fixed segment 1 of FAOL control str
21 3C 37 32 21 20 3D 20 3E 21 53 0035
32 31 21 2F 3E 21 00000048'010E0000' 0040 107 CS_SEG2: .ASCID \!>/!12<!AS!> = !\ ; Fixed segment 2 of FAOL control str
21 20 3D 20 3E 21 53 41 21 3C 004E
3C 39 21 2F 00000060'010E0000' 0058 108 CS_SEG3: .ASCID \!/9<\ ; Fixed segment 3 of FAOL control str
3E 21 53 41 21 0000006C'010E0000' 0064 109 CS_SEG4: .ASCID \!AS!>\ ; Fixed segment 4 of FAOL control str
53 41 21 20 3D 20 00000079'010E0000' 0071 110 CS_SEG5: .ASCID \ = !AS\ ; Fixed segment 5 of FAOL control str
21 3C 34 31 21 2F 00000087'010E0000' 007F 111 CS_SEG6: .ASCID \!/14<!AS!> = !\ ; Fixed segment 6 of FAOL control str
21 20 3D 20 3E 21 53 41 008D
20 20 20 20 20 20 0000009D'010E0000' 0095 112 CS_SEG7: .ASCID \ !AS\ ; Fixed segment 7 of FAOL control str
53 41 21 20 20 20 20 20 20 00A3
00AD 113
41 21 3C 36 32 21 000000B5'010E0000' 00AD 114 CL_SEG1: .ASCID \!26<!AC!>\ ; Fixed seg 1 of classes FAOL ctrl str
3E 21 43 00BB
41 21 3C 36 32 21 000000C6'010E0000' 00BE 115 CL_SEG2: .ASCID \!26<!AC/!AS!>\ ; Fixed seg 2 of classes FAOL ctrl str
3E 21 53 41 21 2F 43 00CC
65 73 73 61 6C 43 000000DB'010E0000' 00D3 116 CLASS_HDG: .ASCID \Classes:\ ; Heading line for classes
3A 73 00E1
65 73 73 61 6C 43 000000EB'010E0000' 00E3 117 NO_CLASS_HDG: .ASCID \Classes: none\ ; Heading line for 'no classes'
65 6E 6F 6E 20 3A 73 00F1
00F8 118
00F8 119 SHOW_FAB: $FAB, - ; FAB for $PARSE to show filespecs
00F8 120 FOP=NAM,-
00F8 121 NAM=SHOW_NAM
0148 122 SHOW_NAM: $NAM, - ; NAM for $PARSE
0148 123 ESA=SHOW_FILESPEC,-
0148 124 ESS=NAM$C_MAXRSS,-
0148 125 NOP=SYNCHR ; syntax check only (don't open file)
01A8 126
000002A7 01A8 127 SHOW_FILESPEC: .BLKB NAM$C_MAXRSS ; space for expanded filespec
02A7 128
000000FF 02A7 129 SHOW_SPEC_D: .LONG - SHOW_FILESPEC ; descriptor for expanded filespec
000001A8' 02AB 130 .LONG SHOW_FILESPEC
02AF 131
00000000 02AF 132 ERROR_QUAL: .LONG 0 ; address of qualifier for filespec
02B3 133 ; which contains a syntax error.
02B3 134
```



```
02B3 136 .SBTTL SHODEF CMD - MONITOR SHOW DEFAULT command
00000000 137 .PSECT $$MONCODE,NOWRT,EXE
0000 138 :++
0000 139 :
0000 140 : FUNCTIONAL DESCRIPTION:
0000 141 :
0000 142 : This routine uses the SCRPKG to display lines in response
0000 143 : to a SHOW subcommand. All qualifiers and their current values
0000 144 : are shown, as well as all selected classes.
0000 145 :
0000 146 : INPUTS:
0000 147 :
0000 148 : None
0000 149 :
0000 150 : IMPLICIT INPUTS:
0000 151 :
0000 152 : SCRDC - quadword string descriptor for buffer required by SCRPKG.
0000 153 : CURR_MRBPTR - pointer to the "current" MRB (Monitor Request Block).
0000 154 : QUALPTR - pointer to the Qualifier Descriptors block.
0000 155 : INTERVAL_DEFAULT - default value for /INTERVAL qualifier.
0000 156 : ALLCL_INT_DEFAULT - default value for /INTERVAL qualifier for ALL class.
0000 157 : VIEWING_DEFAULT - default value for /VIEWING_TIME qualifier.
0000 158 : MAX_CLASS_NO - highest MONITOR class number.
0000 159 :
0000 160 : OUTPUTS:
0000 161 :
0000 162 : None
0000 163 :
0000 164 : IMPLICIT OUTPUTS:
0000 165 :
0000 166 : SHOW command display is sent to the terminal.
0000 167 :
0000 168 : ROUTINE VALUE:
0000 169 :
0000 170 : R0 = $$$_NORMAL, or called routine error status
0000 171 :
0000 172 : SIDE EFFECTS:
0000 173 :
0000 174 : none
0000 175 :
0000 176 : REGISTER USAGE:
0000 177 :
0000 178 : R0,R1,R2,R4,R5 = scratch, used by MOV3
0000 179 : R3 = FAOL control string index
0000 180 : R7 = pointer to MRB (Monitor Request Block)
0000 181 : R8 = pointer to Qualifier Descriptors
0000 182 : R9 = FAOL parameter list index
0000 183 : R10 = address of descriptor for FAOL parameter list
0000 184 : R11 = address of descriptor for FAOL control string
0000 185 :
0000 186 : --
0000 187 :
```

```
00000000'EF 00000000'8F 000002AF'EF 00000000'GF 00000000'EF 00000000'GF 00000000'EF 00000000'EF 0000002F'EF 00000000'EF 0000000F'EF
OFFC 0000 189
0000 190 .ENTRY SHODEF_CMD, ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
0002 191
0002 192
0002 193 MOVL #MNR$_SHOWERR,CURR_ERRCODE ; Set up signaled error code
D4 000D 194 CLRL ERROR_QUAL ; clear address of bad qualifier
0013 195
0013 196 ; Set up a SCRPKG buffered output stream directed to SYS$OUTPUT
0013 197
0013 198
0013 199 PUSHL #0 ; Stack stream identifier
0015 200 CALLS #1,G^SCR$SET_OUTPUT ; Establish output stream
E8 001C 201 BLBS R0,10$ ; Branch if status OK
31 001F 202 BRW SHD_ERR ; Else go exit with error
0022 203
0022 204 10$:
0022 205 PUSHAQ SCRDC ; Push this routine's buffer addr
0028 206 CALLS #1,G^LIB$SET_BUFFER ; Set buffering mode
E8 002F 207 BLBS R0,20$ ; Branch if status OK
31 0032 208 BRW SHD_ERR ; Else go exit with error
0035 209
0035 210 20$:
0035 211 ALLOC 40,R10,R9 ; Allocate an FAOL parameter list
0042 212 ALLOC 80,R11,R3 ; Allocate an FAOL control string
57 0057 213 MOVL CURR_MRBPTR,R7 ; Load up ptr to MRB
58 005E 214 MOVL QUALPTR,R8 ; ... and ptr to qualifier descriptors
0065 215
0065 216 ; Show /BEGINNING qualifier
0065 217
0065 218
0065 219
0065 220 MOVW #^A\!/\,(R3)+ ; Move 'new-line' to FAOL control string
83 006A 221 MOVW CS_SEG1,CS_SEG1+8,(R3) ; Move fixed segment into control string
00000027'EF 89 68 DE 0076 222 MOVAL QUAL$L_BEGTR8),(R9)+ ; /BEGINNING qual name to FAOL prmlst
50 67 7D 0079 223 MOVQ MRB$Q_BEGINNING(R7),R0 ; Test /BEGINNING defaulted?
0A 13 007C 224 BEQL 30$ ; Branch if so
83 4425 8F B0 007E 225 MOVW #^A/%D/,(R3)+ ; FAOL date-time directive to ctrstr
89 67 DE 0083 226 MOVAL MRB$Q_BEGINNING(R7),(R9)+ ; BEGINNING time to FAOL prmlst
1A 11 0086 227 BRB SHO_INT ; Go set up /INTERVAL
0088 228 30$:
83 4341 8F B0 0088 229 MOVW #^A/AC/,(R3)+ ; FAOL cstring directive to ctrstr
1C A7 D5 008D 230 TSTL MRB$A_INPUT(R7) ; Live or Playback?
09 13 0090 231 BEQL 40$ ; Go do live
89 00000000'EF DE 0092 232 MOVAL RV_STR,(R9)+ ; Playback -- cstring ptr to FAOL prmlst
07 11 0099 233 BRB SHO_INT ; Go set up /INTERVAL
009B 234 40$:
89 0000000F'EF DE 009B 235 MOVAL CT_STR,(R9)+ ; Live
; "Current time" cstring ptr to prmlst
```



```
00A2 237 :  
00A2 238 : Show /INTERVAL qualifier  
00A2 239 :  
00A2 240 :  
00A2 241 SHO_INT:  
63 00000048'EF 00000040'EF 28 00A2 242 MOV C3 CS_SEG2,CS_SEG2+8,(R3) ; Move fixed segment into control string  
89 10 A8 DE 00AE 243 MOVAL QUAL$$_INT(R8),(R9)+ ; /INTERVAL qual name to FAOL prmlst  
10 A7 D5 00B2 244 TSTL MRB$_INTERVAL(R7) ; /INTERVAL defaulted ?  
0B 13 00B5 245 BEQL 10$ ; Branch if so  
83 4C5A 8F B0 00B7 246 MOVW #^A/ZL/,(R3)+ ; FAO decimal directive to ctrstr  
89 10 A7 D0 00BC 247 MOVL MRB$_INTERVAL(R7),(R9)+ ; INTERVAL value to FAOL prmlst  
2D 11 00C0 248 BRB 30$ ; Continue.....  
00C2 249 10$: ; /INTERVAL defaulted  
1C A7 D5 00C2 250 TSTL MRB$_INPUT(R7) ; Live or Playback ?  
0E 13 00C5 251 BEQL 20$ ; Go do live  
83 4341 8F B0 00C7 252 MOVW #^A/AC/,(R3)+ ; FAO cstring directive to ctrstr  
89 00000000'EF 1A 11 00CC 253 MOVAL RV_STR,(R9)+ ; Playback -- cstring ptr to FAOL prmlst  
1A 11 00D3 254 BRB 30$ ; Continue.....  
00D5 255 20$: ; Live  
83 4C5A 8F B0 00D5 256 MOVW #^A/ZL/,(R3)+ ; FAO decimal directive to ctrstr  
09 43 A7 0A E0 00DA 257 BBS #MRB$_V_ALL_CLASS,MRB$_V_FLAGS(R7),25$ ; Special default for ALL  
89 00000000'8F 07 11 00DF 258 MOVL #INTERVAL_DEFAULT,(R9)+ ; Default INTERVAL value to prmlst  
07 11 00E6 259 BRB 30$ ; Continue.....  
00E8 260 25$:  
89 00000000'8F D0 00E8 261 MOVL #ALLCL_INT_DEFAULT,(R9)+ ; Default INTERVAL value for ALL class  
00EF 262  
FC A9 DD 00EF 263 30$:  
00EF 264 PUSHL -4(R9) ; Save interval val for /VIEWING_TIME  
00F2 265  
00F2 266 :  
00F2 267 : Display a line showing /BEGINNING and /INTERVAL  
00F2 268 :  
00F2 269 :  
03FB 30 00F2 270 BSBW SHOW SINGLE ; Show the line, single-spaced  
03 50 E8 00F5 271 BLBS RO,SHO_END ; Go on to /ENDING if status OK  
02EF 31 00F8 272 BRW SHD_ERR ; Otherwise, go exit
```



```
00FB 274 :  
00FB 275 : Show /ENDING qualifier  
00FB 276 :  
00FB 277 :  
00FB 278 SHO_END:  
53 04 AB D0 00FB 279 MOVL 4(R11),R3 : Point to beginning of FAOL ctrl string  
59 04 AA D0 00FF 280 MOVL 4(R10),R9 : Point to beginning of FAOL parm list  
0103 281 :  
63 0000002F'EF 00000027'EF 28 0103 282 MOVCL CS_SEG1,CS_SEG1+8,(R3) : Move fixed segment into control string  
89 08 A8 DE 010F 283 MOVAL QUALSL_END(R8),(R9)+ : /ENDING qual name to FAOL prmlst  
50 08 A7 7D 0113 284 MOVQ MRBSQ_ENDING(R7),R0 : Test /ENDING defaulted ?  
0B 13 0117 285 BEQL 10$ : Branch if so  
83 4425 8F B0 0119 286 MOVW #^A/XD/,(R3)+ : FAOL date-time directive to ctrstr  
89 08 A7 DE 011E 287 MOVAL MRBSQ_ENDING(R7),(R9)+ : ENDING time to FAOL prmlst  
1A 11 0122 288 BRB SHO_VIEW : Go set up /VIEWING_TIME  
0124 289 10$: : /ENDING defaulted  
83 4341 8F B0 0124 290 MOVW #^A/AC/,(R3)+ : FAOL cstring directive to ctrstr  
1C A7 D5 0129 291 TSTL MRBSA_INPUT(R7) : Live or Playback ?  
09 13 012C 292 BEQL 20$ : Go do live  
89 00000000'EF DE 012E 293 MOVAL RV_STR,(R9)+ : Playback -- cstring ptr to FAOL prmlst  
07 11 0135 294 BRB SHO_VIEW : Go set up /VIEWING_TIME  
0137 295 20$: : Live  
89 0000001C'EF DE 0137 296 MOVAL ID_STR,(R9)+ : "Indefinite" cstring ptr to prmlst  
013E 297 :  
013E 298 : Show /VIEWING_TIME qualifier  
013E 299 :  
013E 300 :  
013E 301 :  
63 00000048'EF 00000040'EF 28 013E 302 SHO_VIEW:  
89 20 A8 DE 014A 303 MOVCL CS_SEG2,CS_SEG2+8,(R3) : Move fixed segment into control string  
83 4C5A 8F B0 014E 304 MOVAL QUALSL_VIEW(R8),(R9)+ : Qualifier name to FAOL prmlst  
18 A7 D5 0153 305 MOVW #^A/ZL7,(R3)+ : FAOL decimal directive to ctrstr  
06 13 0156 306 TSTL MRBSL_VIEWING_TIME(R7) : /VIEWING TIME defaulted ?  
89 18 A7 D0 0158 307 BEQL 10$ : Branch if so  
11 11 015C 308 MOVL MRBSL_VIEWING_TIME(R7),(R9)+ : VIEWING_TIME value to FAOL prmlst  
015E 309 BRB 30$ : Continue.....  
1C A7 D5 015E 310 10$: : /VIEWING TIME defaulted  
09 13 0161 311 TSTL MRBSA_INPUT(R7) : Live or Playback ?  
89 00000000'8F D0 0163 312 BEQL 20$ : Go do live  
03 11 016A 313 MOVL #VIEWING_DEFAULT,(R9)+ : Default VIEWING_TIME value to prmlst  
016C 314 BRB 30$ : Continue.....  
89 8E D0 016C 315 20$: : Live  
016F 316 MOVL (SP)+,(R9)+ : Pop saved /INTERVAL value to prmlst  
016F 317 30$: :  
016F 318 :  
016F 319 : Display a line showing /ENDING and /VIEWING_TIME  
016F 320 :  
016F 321 :  
016F 322 :  
0382 30 016F 323 BSBW SHOW DOUBLE : Show the line, double-spaced  
03 50 E8 0172 324 BLBS RO,SHO_FLUSH : Go on to file qualifiers if status OK  
0272 31 0175 325 BRW SHD_ERR : Otherwise, go exit  
0178 326
```

```
0178 328 ;
0178 329 ; Show /FLUSH_INTERVAL qualifier
0178 330 ;
0178 331 ;
0178 332 SHO_FLUSH:
53 04 AB D0 0178 333 MOVL 4(R11),R3 ; Point to beginning of FAOL ctrl string
59 04 AA D0 017C 334 MOVL 4(R10),R9 ; Point to beginning of FAOL parm list
0180 335
63 00000087'EF 0000007F'EF 28 0180 336 MOVCL CS_SEG6,CS_SEG6+8,(R3) ; Move fixed segment into control string
89 18 A8 DE 018C 337 MOVAL QUAL$ _FLUSH(R8),(R9)+ ; Qualifier name to FAOL prmlst
83 4C5A 8F B0 0190 338 MOVW #^A/ZL7,(R3)+ ; FAOL decimal directive to ctrstr
14 A7 D5 0195 339 TSTL MRB$_FLUSH(R7) ; /FLUSH defaulted ?
06 13 0198 340 BEQL 10$ ; Branch if so
89 14 A7 D0 019A 341 MOVL MRB$_FLUSH(R7),(R9)+ ; FLUSH value to FAOL prmlst
07 11 019E 342 BRB 30$ ; Continue....
89 00000000'8F D0 01A0 343 10$: ; /FLUSH defaulted
01A0 344 MOVL #FLUSH_INT_DEFAULT,(R9)+ ; Default FLUSH value to prmlst
01A7 345 30$:
01A7 346
01A7 347 ; Display a line showing /FLUSH_INTERVAL
01A7 348 ;
01A7 349 ;
034A 30 01A7 351 BSBW SHOW DOUBLE ; Show the line, double-spaced
03 50 E8 01AA 352 BLBS RO,SHO_FILES ; Go on to file qualifiers if status OK
023A 31 01AD 353 BRW SHD_ERR ; Otherwise, go exit
01B0 354
01B0 355 ;
01B0 356 ; Show qualifiers which always have string values
01B0 357 ; (if they are present). These are typically qualifiers
01B0 358 ; with file specs as values.
01B0 359 ;
01B0 360
01B0 361 SHO_FILES:
01B0 362
01B0 363 ALLOC 8,R0,R6 ; Allocate a pair of longwords to pass
01BD 364 ; ... as input parameter in R6 to
01BD 365 ; ... SHOW_FILE_QUAL
01BD 366 ;
01BD 367 ; At this point, MRB$_INPUT contains the address of the IFB table, or 0
01BD 368 ; if there is no current default. (Later, MRB$_INPUT will be changed
01BD 369 ; to be the address of a single file spec descriptor, unless we are
01BD 370 ; doing a multi-file summary.)
01BD 371 ;
04 66 1C A7 D0 01BD 372 MOVL MRB$_INPUT(R7),(R6) ; Load addr of qualifier value descr
04 A6 28 A8 DE 01C1 373 MOVAL QUAL$_INP(R8),4(R6) ; Load addr of qualifier name descr
0045 30 01C6 374 BSBW SHOW_INPUT_QUAL ; display input qualifier
3F 50 E9 01C9 375 BLBC RO,SF_ERR ; Go return if error
01CC 376
04 66 24 A7 D0 01CC 377 MOVL MRB$_RECORD(R7),(R6) ; Load addr of qualifier value descr
04 A6 38 A8 DE 01D0 378 MOVAL QUAL$_REC(R8),4(R6) ; Load addr of qualifier name descr
00D6 30 01D5 379 BSBW SHOW_FILE_QUAL ; Show a line for /RECORD
30 50 E9 01D8 380 BLBC RO,SF_ERR ; Go return if error
01DB 381
04 66 20 A7 D0 01DB 382 MOVL MRB$_DISPLAY(R7),(R6) ; Load addr of qualifier value descr
04 A6 30 A8 DE 01DF 383 MOVAL QUAL$_DISP(R8),4(R6) ; Load addr of qualifier name descr
00C7 30 01E4 384 BSBW SHOW_FILE_QUAL ; Show a line for /DISPLAY
```



```
21 50 E9 01E7 385 BLBC R0,SF_ERR ; Go return if error
      01EA 386
04 66 28 A7 D0 01EA 387 MOVL MRB$A_SUMMARY(R7),(R6) ; Load addr of qualifier value descr
A6 40 A8 DE 01EE 388 MOVAL QUAL$C_SUMM(R8),4(R6) ; Load addr of qualifier name descr
      00B8 30 01F3 389 BSBW SHOW_FILE_QUAL ; Show a line for /SUMMARY
      12 50 E9 01F6 390 BLBC R0,SF_ERR ; Go return if error
      01F9 391
04 66 2C A7 D0 01F9 392 MOVL MRB$A_COMMENT(R7),(R6) ; Load addr of qualifier value descr
A6 48 A8 DE 01FD 393 MOVAL QUAL$C_COMM(R8),4(R6) ; Load addr of qualifier name descr
      00FF 30 0202 394 BSBW SHOW_QUAL ; Show a line for /COMMENT
      03 50 E9 0205 395 BLBC R0,SF_ERR ; Branch on error
      013D 31 0208 396 BRW SCAN_CLASSES ; Go on to show classes if no errors
      020B 397
      020B 398 SF_ERR:
01DC 31 020B 399 BRW SHD_ERR ; Go log error and return
```



```
63 00000060'EF 00000058'EF 58 DD 020E 401 SHOW_INPUT_QUAL:
      53 58 01 D0 020E 402 PUSH R8 ; save R8 so we can use it as scratch
      59 04 AB D0 0210 403 MOVL #1,R8 ; init input file counter
      66 04 AA D0 0213 404 MOVL 4(R11),R3 ; Point to beginning of FAOL ctrl string
      05 05 05 D0 0217 405 MOVL 4(R10),R9 ; Point to beginning of FAOL parm list
      83 4F4E 8F B0 021B 406 MOV C3 CS_SEG3,CS_SEG3+8,(R3) ; Move fixed segment into control string
      05 12 0227 407 TSTL (R6) ; Qualifier present?
      05 12 0229 408 BNEQ 5$ ; Branch if yes
      83 4F4E 8F B0 022B 409 MOVW #^A/NO/, (R3)+ ; No -- move NO to control string
63 0000006C'EF 00000064'EF 28 0230 410 5$:
      89 04 A6 D0 0230 411 MOV C3 CS_SEG4,CS_SEG4+8,(R3) ; Move fixed segment into control string
      66 05 023C 412 MOVL 4(R6),(R9)+ ; Qualifier name descr to FAOL prmlst
      02 12 0240 413 TSTL (R6) ; Qualifier present?
      19 11 0242 414 BNEQ 7$ ; Branch if yes
      55 42 A7 9A 0244 415 BRB 10$ ; No, go straight to SHOW_DOUBLE call
      01 55 D1 0246 416 7$:
      16 14 0246 417 MOVZBL MRBSB_INP_FILES(R7),R5 ; Number of input files to R5
      024A 418 CMPL R5,#1 ; How many input files?
      024D 419 BGTR 12$ ; branch if there are > 1 input files
      024F 420 ;
      024F 421 ; If we got here, only one file was specified for /INPUT, so just
      024F 422 ; do a SHOW_DOUBLE for that file and the /INPUT qualifier, and get out.
      024F 423 ;
63 00000079'EF 00000071'EF 28 024F 424 MOV C3 CS_SEG5,CS_SEG5+8,(R3) ; Move fixed segment into control string
      89 00 B6 D0 025B 425 MOVL @ (R6),(R9)+ ; Qual value descr to FAOL prmlst
      025F 426 10$:
      025F 427 ;
      025F 428 ;
      025F 429 ; At this point, either 0 (/NOINPUT case) or 1 file was specified for input.
      025F 430 ;
      025F 431 ;
      0292 30 025F 432 BSBW SHOW_DOUBLE ; Show the line, double-spaced
      0045 31 0262 433 BRW 25$ ; get out
      0265 434 ;
      0265 435 ;
      0265 436 ; Begin multi-file summary loop
      0265 437 ;
      0265 438 12$:
      01 55 DD 0265 439 PUSH R5 ; Save count of # of input files
      58 D1 0267 440 CMPL R8,#1 ; Are we on file #1?
      0E 12 026A 441 BNEQ 15$ ; Branch if we have passed file #1
      026C 442 ;
      026C 443 ; If we got here, we are doing a multi-file summary and we are processing
      026C 444 ; input file #1.
      026C 445 ;
63 00000079'EF 00000071'EF 28 026C 446 MOV C3 CS_SEG5,CS_SEG5+8,(R3) ; Move fixed segment into control string
      0C 11 0278 447 BRB 18$ ; skip past alternate control string
      027A 448 15$:
      027A 449 ;
      027A 450 ; If we got here, this is a multi-file summary
      027A 451 ; and we have processed the first file in the list already,
      027A 452 ; so we use a different FAOL control string
      027A 453 ;
63 0000009D'EF 00000095'EF 28 027A 454 MOV C3 CS_SEG7,CS_SEG7+8,(R3) ; Move fixed segment into control string
      55 8ED0 0286 455 18$:
      89 00 B6 D0 0286 456 POPL R5 ; get R5 back (pushed to save from MOV C)
      0289 457 MOVL @ (R6),(R9)+ ; Qual value descr to FAOL prmlst
```

```
66 05 C0 028D 458 ADDL2 #IFB$K_SIZE,(R6) ; move to next IFB
      0290 459
      0290 460 20$:
58 55 D1 0290 461 CMPL R5,R8 ; Is this the last input file?
      05 12 0293 462 BNEQ 23$ ; branch if not
      025C 30 0295 463 BSBW SHOW_DOUBLE ; Show the last line, double-spaced
      10 11 0298 464 BRB 25$ ; and get out
      0253 30 029A 465 23$:
53 04 AB D0 029D 466 BSBW SHOW_SINGLE ; Show the line, single-spaced
59 04 AA D0 02A1 467 MOVL 4(R1T),R3 ; Point to beginning of FAOL ctrl string
      58 D6 02A5 468 MOVL 4(R10),R9 ; Point to beginning of FAOL parm list
      FFBB 31 02A7 469 INCL R8 ; on to next input file.
      02AA 470 BRW 12$ ; Loop
      02AA 471
      02AA 472 25$:
      58 8ED0 02AA 473 POPL R8 ; Restore R8
      05 02AD 474 RSB ; Return with status in R0
```



```

66  D5 02AE 476 SHOW_FILE QUAL:
52  13 02AE 477 TSTL (R6) ; Qualifier present?
    02B0 478 BEQL SHOW_QUAL ; Branch if not
    02B2 479
    02B2 480
    02B2 481 : Use $PARSE to get filespec with defaults applied
    02B2 482 : and password removed from access control string. Note that the file
    02B2 483 : does not exist yet, but $PARSE will still do the work of
    02B2 484 : applying defaults and removing the password, if necessary
    02B2 485 :
    50 01 D0 02B2 486 MOVL #1,R0 ; set up for post-indexing
    00 B6 90 02B5 487 MOVBL @ (R6),-
00000124'EF 00 B640 D0 02B8 488 SHOW_FAB + FAB$B_FNS ; plug FAB with filespec length
    02BD 489 MOVL @ (R6)[R0],-
    02C6 490 SHOW_FAB + FAB$B_FNA ; plug FAB with filespec address
    02C6 491 $PARSE FAB=SHOW_FAB ; do the parse
    1C 50 E8 02D3 492 BLBS R0,10$ ; Branch if OK
00000000'8F 50 D1 02D6 493 CMPL R0,#RMS$_SYN ; Filespec syntax error?
    01 13 02DD 494 BEQL 5$ ; Branch if so
    05 02DF 495 RSB ; unknown error, return with status
    02E0 496
    02E0 497 : file specification syntax error processing: save the bad qualifier name,
    02E0 498 : to be reported after SHOW display finishes. Note that we can only
    02E0 499 : report on one syntax error per SHOW.
    02E0 500
    000002AF'EF D5 02E0 501 5$: TSTL ERROR_QUAL ; has there already been a syntax error?
    1C 12 02E6 502 BNEQ SHOW_QUAL ; branch if so (we can only report first err
000002AF'EF 04 A6 D0 02E8 503 MOVL 4(R6),- ; load address of offending qualifier
    12 11 02F0 504 ERROR_QUAL
    02F2 505 BRB SHOW_QUAL ; display unparsed spec containing error
    00000153'EF 9A 02F2 506 10$: MOVZBL SHOW_NAM+NAM$B_ESL,- ; Plug descriptor with length
    000002A7'EF 02F8 507 SHOW_SPEC_D ; passed back from $PARSE
66 000002A7'EF DE 02FD 508 MOVAL SHOW_SPEC_D,(R6) ; point to parsed spec descriptor
    0304 509 ; and fall through to SHOW_QUAL
    0304 510 SHOW_QUAL:
    0304 511
    53 04 AB D0 0304 512 MOVL 4(R11),R3 ; Point to beginning of FAOL ctrl string
    59 04 AA D0 0308 513 MOVL 4(R10),R9 ; Point to beginning of FAOL parm list
63 00000060'EF 00000058'EF 28 030C 514 MOVCL CS_SEG3,CS_SEG3+8,(R3) ; Move fixed segment into control string
    66 D5 0318 515 TSTL (R6) ; Qualifier present?
    05 12 031A 516 BNEQ 10$ ; Branch if yes
    83 4F4E 8F B0 031C 517 MOVW #^A/NO/, (R3)+ ; No -- move NO to control string
    0321 518 10$: MOVCL CS_SEG4,CS_SEG4+8,(R3) ; Move fixed segment into control string
63 0000006C'EF 00000064'EF 28 0321 519 MOVL 4(R6),(R9)+ ; Qualifier name descr to FAOL prmlst
    89 04 A6 D0 032D 520 TSTL (R6) ; Qualifier present?
    66 D5 0331 521 BEQL 20$ ; Branch if no
    0F 13 0333 522 MOVCL CS_SEG5,CS_SEG5+8,(R3) ; Move fixed segment into control string
63 00000079'EF 00000071'EF 28 0335 523 MOVL (R6),(R9)+ ; Qual value descr to FAOL prmlst
    89 66 D0 0341 524
    0344 525
    0344 526
    0344 527
    0344 528 : Display a line showing current qualifier
    0344 529
    0344 530
    0344 531
    0344 532 20$:

```


I 14

- MONITOR SHOW DEFAULT Command	16-SEP-1984 02:05:00	VAX/VMS Macro V04-00
SHODEF_CMD - MONITOR SHOW DEFAULT comman	5-SEP-1984 02:02:35	[MONITOR.SRC]SHODEF.MAR;1

Page 14
(11)

```

01AD 30 0344 533      BSBW  SHOW_DOUBLE      ; Show the line, double-spaced
      0347 534
      05 0347 535      RSB      ; Return with status in R0

```

```
0348 537 SCAN_CLASSES:
0348 538
0348 539 :
0348 540 : Scan the MRB$O_CLASSBITS field of the MRB to determine the selected
0348 541 : classes. Create a vector of bytes containing a class number for each
0348 542 : selected class. Then show the selected classes by calling SHOW_CLASSES
0348 543 : with the vector as input.
0348 544 :
0348 545 :
0348 546 : REGISTER USAGE:
0348 547 :
0348 548 : R0,R1 = scratch
0348 549 : R2 = bit field starting position for FFS instruction
0348 550 : R3 = bit field size for FFS instruction
0348 551 : R4 = class number determined by FFS instruction
0348 552 : R5 = index into MRB$O_CLASSBITS
0348 553 : R6 = pointer to byte vector of class numbers
0348 554 : R7 = pointer to MRB (Monitor Request Block)
0348 555 : R8 = pointer to Qualifier Descriptors
0348 556 : R9 = class counter
0348 557 : R10 = address of descriptor for FAOL parameter list
0348 558 : R11 = address of descriptor for FAOL control string
0348 559 :
0348 560 :
0348 561 : ALLOC MAX_CLASS_NO+1,R0,R6 ; Get a byte vector for class numbers
0348 562 : ; to be shown, and point R6 to it
59 D4 035D 563 : CLRL R9 ; Init class counter
035F 564 :
035F 565 :
035F 566 : Use FFS instruction to scan the class bits in the MRB, filling
035F 567 : the above byte vector with a class number for each class found.
035F 568 : R9 contains the count of classes found.
035F 569 :
035F 570 :
55 D4 035F 571 : CLRL R5 ; Init starting bit position
0361 572 20$: :
53 20 D0 0361 573 : MOVL #32,R3 ; Init bit field size
0364 574 : ; NOTE -- must handle in 32-bit chunks
52 55 D0 0364 575 : MOVL R5,R2 ; Init start position of next chunk
0367 576 30$: :
54 32 A7 53 52 EA 0367 577 : FFS R2,R3,MRB$O_CLASSBITS(R7),R4 ; Search for next class number
036D 578 : ; R4 contains class no. if found
036D 579 : BEQL 40$ ; Branch if none found this chunk
6649 54 90 036F 580 : MOV B R4,(R6)[R9] ; Move class no. into byte vector
59 D6 0373 581 : INCL R9 ; Count this class
0375 582 :
0375 583 : ADDL2 R2,R3 ; Compute next starting
52 54 01 C1 0378 584 : ; ... position and field size
53 52 C2 037C 585 : ; ... for this chunk
E6 11 037F 586 : BRB 30$ ; Go search rest of chunk
FFD8 55 20 0000'8F 3D 0381 587 40$: :
0381 588 : ACBW #MAX_CLASS_NO,#32,R5,20$ ; Loop to process next chunk
```



```

0389 590 :
0389 591 : At this point, R6 points to a byte vector consisting of the
0389 592 : class numbers for classes to show. R9 contains the count of
0389 593 : classes to show.
0389 594 :
0389 595 :
0389 596 :
0389 597 : Show class-names and their associated display qualifiers.
0389 598 :
0389 599 :
0389 600 : PUSHL R11 ; Stack addr of descr for FAOL ctrl str
0389 601 : PUSHL R10 ; Stack addr of descr for FAOL prm list
0389 602 : PUSHL R6 ; Stack addr of byte vector of class nos.
0389 603 : PUSHL R9 ; Stack count of classes to show
0000041E'EF 04 FB 0391 604 : CALLS #4,SHOW_CLASSES ; Show all classes and their
4F 50 E9 0398 605 : ; associated display qualifiers
0398 606 : BLBC R0,SHD_ERR ; Exit if error
0398 607 :
0398 608 :
0398 609 : Print one final blank line for readability
0398 610 :
0398 611 :
0398 612 : ALLOC 8,R0,R1 ; Get quadword for dummy descr
04 A1 04 A1 D4 03A8 613 : CLRL (R1) ; Make length 0
01 DD 03AA 614 : MOVAL 4(R1),4(R1) ; ... and point it to itself
00000000'GF 02 FB 03AF 615 : PUSHL #1 ; Stack single spacing indicator
2D 50 E9 03B1 616 : PUSHL R1 ; ... and dummy text descriptor
03B3 617 : CALLS #2,G^SCR$PUT_LINE ; Put blank line to the terminal
03BA 618 : BLBC R0,SHD_ERR ; Exit if error
03BD 619 :
03BD 620 :
03BD 621 :
03BD 622 : SHOW subcommand processing complete
03BD 623 :
03BD 624 :
03BD 625 :
03BD 626 :
03BD 627 : See if we encountered a filespec syntax error on SHOW_FILE_QUAL parse,
03BD 628 : and if so, report it.
03BD 629 :
03BD 630 :
50 00000000'8F D0 03BD 630 : MOVL #$$$ NORMAL,R0 ; Assume success status
000002AF'EF D5 03C4 631 : TSTL ERROR_QUAL ; was there a syntax error?
36 13 03CA 632 : BEQL SHD_RET ; no, return with success
000002AF'EF DD 03CC 633 : PUSHL ERROR_QUAL ; push address of qualifier descriptor
7E D4 03D2 634 : CLRL -(SP) ; no secondary code
00000000'8F DD 03D4 635 : PUSHL #MNR$ FILSYNERR ; Push syntax error status code
00000000'EF 03 FB 03DA 636 : CALLS #3,MON_ERR
50 00000000'8F D0 03E1 637 : MOVL #MNR$ FILSYNERR,R0 ; Get status to caller
18 11 03E8 638 : BRB SHD_RET ; Go return
03EA 639 :
03EA 640 SHD_ERR: ; Log error and return
50 DD 03EA 641 : PUSHL R0 ; Bad status on stack
6E DF 03EC 642 : PUSHL (SP) ; Stack pointer to bad status
00000000'8F DD 03EE 643 : PUSHL #MNR$ SHOWERR ; Stack MONITOR failing status code
00000000'EF 02 FB 03F4 644 : CALLS #2,MON_ERR ; Log the error
50 00000000'8F D0 03FB 645 : MOVL #MNR$ SHOWERR,R0 ; Get status to caller
0402 646 :

```

[illegible]

SHODEF
V04-000

- MONITOR SHOW DEFAULT Command L 14 16-SEP-1984 02:05:00 VAX/VMS Macro V04-00 Page 17
SHODEF_CMD - MONITOR SHOW DEFAULT comman 5-SEP-1984 02:02:35 [MONITOR.SRC]SHODEF.MAR;1 (13)

		0402	647	SHD_RET:		; Return point from SHODEF_CMD routine
		0402	648			
01	BB	0402	649	PUSHR	#^M<R0>	; Save return status
00000000'GF	00	FB	0404	650		
		0404	651	CALLS	#0,G^LIB\$PUT_BUFFER	; Output SCRPKG buffer & stop buffering
		040B	652			
00	DD	040B	653	PUSHL	#0	; Indicate "clear buffer mode"
00000000'GF	01	FB	040D	654	CALLS	#1,G^LIB\$SET_BUFFER
00000000'GF	00	FB	0414	655	CALLS	#0,G^SCR\$STOP_OUTPUT
		041B	656			; Stop output stream
01	BA	041B	657	POPR	#^M<R0>	; Get back SHODEF_CMD return status
	04	041D	658	RET		; Return with status in R0

SHO
Sym

RMS
RV
SAC
SCA
SCR
SCR
SCR
SF
SHC
SHC
SHD
SHD
SHO
SHO
SHO
SHO
SHO
SHO
SHO
SHO
SHO
SHO
SHO
SHO
SHO
SS\$
STA
STA
SYS
SYS
SYS
TOP
TOP
TOP
TOP
VIE

PSE

MON
SAE
SS\$


```
041E 660 .SBTTL SHOW_CLASSES - Show all selected classes
041E 661
041E 662 :++
041E 663 :
041E 664 : FUNCTIONAL DESCRIPTION:
041E 665 :
041E 666 : SHOW_CLASSES fills out the FAOL parameter list and
041E 667 : control string with the information required to
041E 668 : display each of the classes followed by the display
041E 669 : qualifier for each. It accepts as input a count of
041E 670 : the number of classes to show and a byte vector
041E 671 : containing a class number for each class.
041E 672 :
041E 673 : INPUTS:
041E 674 :
041E 675 : 4(AP) - count of classes to be shown
041E 676 :
041E 677 : 8(AP) - address of byte vector containing a class number
041E 678 : for each selected class
041E 679 :
041E 680 : 12(AP) - address of descriptor for FAOL parameter list
041E 681 :
041E 682 : 16(AP) - address of descriptor for FAOL control string
041E 683 :
041E 684 : IMPLICIT INPUTS:
041E 685 :
041E 686 : CDBHEAD - table of contiguous CDB's, one for each class.
041E 687 :
041E 688 : CLASSTABLE - table of contiguous quadwords, one for each class.
041E 689 : Each quadword consists of a pointer to a counted
041E 690 : ASCII string for the class name followed by a
041E 691 : longword containing the class number.
041E 692 :
041E 693 : PROCD_TABLE - table of contiguous longword pointers, one for
041E 694 : each PROCESSES display qualifier. Each pointer
041E 695 : points to a string descriptor for the qualifier
041E 696 : name.
041E 697 :
041E 698 : STAT_TABLE - table similar to PROCD_TABLE but instead points
041E 699 : to statistic qualifiers for standard classes.
041E 700 :
041E 701 : OUTPUTS:
041E 702 :
041E 703 : none
041E 704 :
041E 705 : IMPLICIT OUTPUTS:
041E 706 :
041E 707 : FAOL control string and parameter list updated.
041E 708 :
041E 709 : ROUTINE VALUE:
041E 710 :
041E 711 : R0 = $$$_NORMAL, or called routine error status.
041E 712 :
041E 713 : SIDE EFFECTS:
041E 714 :
041E 715 : none
041E 716 :
```

SHODEF
V04-000

- MONITOR SHOW DEFAULT Command N 14
SHOW_CLASSES - Show all selected classes 16-SEP-1984 02:05:00 VAX/VMS Macro V04-00
5-SEP-1984 02:02:35 [MONITOR.SRC]SHODEF.MAR;1

Page 19
(14)

041E 717 :
041E 718 :--
041E 719

**F


```

                                041E 721
                                041E 722 .ENTRY SHOW_CLASSES, ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                                0420 723
5A 0C AC D0 0420 724          MOVL 12(AP),R10          ; Load ptr to descr for FAOL parm list
5B 10 AC D0 0424 725          MOVL 16(AP),R11         ; Load ptr to descr for FAOL ctrl str
                                0428 726
                                0428 727          TSTL 4(AP)          ; Test number of classes
                                03 12 042B 728          BNEQ 10$        ; Branch if at least one
                                004B 31 042D 729          BRW 50$        ; None -- go tell user
                                0430 730
                                0430 731 ;
                                0430 732 ; Display "Classes:" heading
                                0430 733 ;
                                0430 734
                                0430 735 10$:
                                0430 736          PUSHL #1          ; Stack "single-spacing" indicator
                                0432 737          PUSHAL CLASS_HDG    ; ... and text descriptor
                                0438 738          CALLS #2,G^$CR$PUT_LINE ; Put header to the terminal
                                043F 739          BLBC R0,SHC_RET      ; Exit if error
                                0442 740
                                0442 741          CLRL R7             ; Init byte vector index
                                0444 742          CLRL R8             ; Indicate this class at beg of line
                                0446 743 20$:
                                0446 744          TSTL R8             ; Need to start this class on new line?
                                0448 745          BNEQ 30$            ; Branch if not
                                044A 746
                                044A 747          MOVL 4(R10),R9        ; Point to beginning of FAOL parm list
                                044E 748          MOVL 4(R11),R3       ; Point to beginning of FAOL ctrl string
                                0452 749          MOVB #^A/ /,(R3)+    ; Start classes in column 2 of screen
                                0455 750          MOVL #3,R8           ; Init "classes per line" count
                                0458 751 30$:
                                0458 752          MOVL 8(AP),R0        ; Get addr of byte vector
                                045C 753          MOVZBL (R0)[R7],R2    ; Get class number for BUILD_FAOL_ARGS
                                0460 754          BSBW BUILD_FAOL_ARGS ; Bld FAOL prmlst & ctrstr 4 this class
                                0463 755
                                0463 756
                                0463 757          DECL R8             ; NOTE -- this rtn destroys REGS R0-R2
                                0465 758          BNEQ 40$            ; ... and R4-R6. Also updates R3 and R9
                                0467 759          ; Update "classes left this line"
                                0467 760          ; Branch if at least one left
                                0467 761 ;
                                0467 762 ; Show a line of 3 classes
                                0467 763 ;
                                0467 764
                                0086 30 0467 764          BSBW SHOW_SINGLE ; Show the line, single-spaced
                                27 50 E9 046A 765          BLBC R0,SHC_RET ; Exit if error
                                046D 766
                                046D 767 40$:
                                046D 767          AOBLS 4(AP),R7,20$    ; Loop back to do next class
```



```
58 D5 0472 769 TSTL R8 ; Classes remaining to be shown?
17 13 0474 770 BEQL SHC_NORM ; Branch if not
    0476 771
    0476 772 ;
    0476 773 ; Show final class line
    0476 774 ;
    0476 775
0077 30 0476 776 BSBW SHOW_SINGLE ; Show the line, single-spaced
12 11 0479 777 BRB SHC_NORM ; ... and return with normal status
    047B 778
    047B 779 ;
    047B 780 ; Show 'No classes' heading
    047B 781 ;
    047B 782
    047B 783 50$:
01 DD 047B 784 PUSHL #1 ; Stack 'single-spacing' indicator
000000E3'EF DF 047D 785 PUSHAL NO_CLASS_HDG ; ... and text descriptor
00000000'GF 02 FB 0483 786 CALLS #2,G^SCR$PUT_LINE ; Put header to the terminal
07 50 E9 048A 787 BLBC R0,SHC_RET ; Exit if error
    048D 788
    048D 789 SHC_NORM:
50 00000000'8F D0 048D 790 MOVL #SS$_NORMAL,R0 ; No failing status hit
    0494 791
    0494 792 SHC_RET:
04 0494 793 RET ; Return with status already in R0
```



```
0495 795 :  
0495 796 : BUILD_FAOL_ARGS subroutine.  
0495 797 :  
0495 798 : This subroutine annexes FAOL directives to the control  
0495 799 : string, and parameters to the parameter list for the  
0495 800 : current class.  
0495 801 :  
0495 802 :  
0495 803 : REGISTER USAGE:  
0495 804 :  
0495 805 :     R0,R1    = scratch  
0495 806 :     R2       = class number of current class (input)  
0495 807 :     R3       = next available byte in FAOL control string (input)  
0495 808 :     R4,R5,R6 = scratch  
0495 809 :     R9       = next available longword in FAOL parameter list (input)  
0495 810 :  
0495 811 :     NOTE -- R3 and R9 are updated. R0-R2 and R4-R6 are destroyed.  
0495 812 :  
0495 813 :  
0495 814 BUILD_FAOL_ARGS:  
0495 815 :  
0495 816 :  
0495 817 : Move class-name cstring pointer to FAOL parameter list  
0495 818 :  
0495 819 :  
50 00000004'EF DE 0495 820      MOVAL  CLASSTABLE+4,R0      ; Get addr of table of class quadwords  
    50 6042 7D 049C 821      MOVQ    (R0)[R2],R0      ; R0 gets class-name cstring ptr  
    89 50  D0 04A0 822      MOVL    R0,(R9)+          ; Move it to FAOL parm list  
04A3 823 :  
04A3 824 :  
04A3 825 : Obtain CDB address for this class  
04A3 826 :  
04A3 827 :  
56 52 00000053 8F C5 04A3 828      MULL3  #CDB$K_SIZE,R2,R6      ; Get CDB offset from class number  
04AB 829 :  
56 00000000'EF46 9E 04AB 830      MOVAB  CDBHEAD[R6],R6      ; NOTE - this rtn no longer needs class num  
04B3 831 :  
04B3 832 :  
04B3 833 : Move appropriate segment of FAO directives into FAO control string  
04B3 834 : and move address of descriptor for display qualifier to FAOL parm list,  
04B3 835 : if one exists.  
04B3 836 :  
04B3 837 :  
    50 44 A6 9A 04B3 838      MOVZBL  CDB$B_ST_CUR(R6),R0      ; R0 gets display qualifier index  
    09 4B A6 E0 04B7 839      BBS     #CDB$V STD,CDB$L_FLAGS(R6),10$ ; Branch if standard class  
56 00000000'EF DE 04BC 840      MOVAL  PROCD_TABLE,R6      ; Get ptr to PROCESSES display qual table  
    07 11 04C3 841      BRB     20$      ; ... and go get desired element  
04C5 842 10$:  
56 00000000'EF DE 04C5 843      MOVAL  STAT_TABLE,R6      ; Get ptr to statistic qualifier ...  
04CC 844 :  
04CC 845 20$:  
    6640 D5 04CC 846      TSTL    (R6)[R0]      ; Is there a qual defined for this stat?  
    12 13 04CF 847      BEQL    30$      ; Branch if no  
    89 6640 D0 04D1 848      MOVL    (R6)[R0],(R9)+      ; Move descr ptr to FAOL prmlst  
63 000000C6'EF 000000BE'EF 28 04D5 849      MOVCL  CL_SEG2,CL_SEG2+8,(R3) ; Move fixed segment into ctrl string  
    0C 11 04E1 850      BRB     40$      ; Go return  
04E3 851
```

SHODEF
V04-000

- MONITOR SHOW DEFAULT Command
SHOW_CLASSES - Show all selected classes

E 15

16-SEP-1984 02:05:00 VAX/VMS Macro V04-00
5-SEP-1984 02:02:35 [MONITOR.SRC]SHODEF.MAR;1

Page 23
(18)

63	000000B5'EF	000000AD'EF	28	04E3	852	30\$:				
				04E3	853		MOV	CL_SEG1,CL_SEG1+8,(R3)	; Move fixed segment into control...	
				04EF	854				; ... string (no display qualifier)...	
				04EF	855	40\$:				
			05	04EF	856		RSB			


```
04F0 858 :  
04F0 859 : SHOW_SINGLE and SHOW_DOUBLE subroutine.  
04F0 860 : The 2 routine names are alternate entry points to show a line with  
04F0 861 : single-spacing and double-spacing, respectively.  
04F0 862 : Upon entry, R10 points to the FAOL parameter list descriptor,  
04F0 863 : R11 points to the FAOL control string descriptor and R3 to the next  
04F0 864 : available byte in the control string. The routine updates the control  
04F0 865 : string descriptor, destroys R1, and returns status in R0.  
04F0 866 :  
04F0 867 :  
01 DD 04F0 868 SHOW_SINGLE: ; Show a line and advance one line  
02 11 04F0 869 PUSHL #1 ; Stack "single-space" indicator  
04F2 870 BRB SHOW_COMM ; Join common code  
04F4 871 :  
02 DD 04F4 872 SHOW_DOUBLE: ; Show a line and advance two lines  
04F4 873 PUSHL #2 ; Stack "double-space: indicator  
04F6 874 :  
50 04 AB D0 04F6 875 SHOW_COMM: ; Common point for 2 entry points  
6B 53 50 C3 04FA 876 MOVL 4(R11),R0 ; Get address of FAOL control string  
04 AA DD 04FE 877 SUBL3 R0,R3,(R11) ; Compute actual length  
6B DF 0501 878 PUSHL 4(R10) ; Stack addr of FAOL parm list  
0000050B'EF 03 FB 0503 879 PUSHAL (R11) ; Stack addr of FAOL ctr str descr  
05 050A 880 CALLS #3,SHOW_A_LINE ; Display one line of SHOW output  
881 RSB ; Return with status in R0
```

```
050B 883 .SBTTL SHOW_A_LINE - Put a line of SHOW to terminal
050B 884
050B 885 :++
050B 886
050B 887 : FUNCTIONAL DESCRIPTION:
050B 888
050B 889 : SHOW_A_LINE sends one display line of SHOW output to the
050B 890 : terminal via the SCRPKG. The line to display is defined
050B 891 : by an $FAOL control string and parameter list, both of
050B 892 : which are input to this routine.
050B 893
050B 894 : INPUTS:
050B 895
050B 896 : 4(AP) - address of descriptor for $FAOL control string.
050B 897
050B 898 : 8(AP) - address of $FAOL parameter list.
050B 899
050B 900 : 12(AP) - number of display lines to advance after showing a line.
050B 901
050B 902 : IMPLICIT INPUTS:
050B 903
050B 904 : OUTDSC - quadword string descriptor for $FAOL output buffer.
050B 905
050B 906 : OUTPUTS:
050B 907
050B 908 : none
050B 909
050B 910 : IMPLICIT OUTPUTS:
050B 911
050B 912 : SHOW line sent to Screen Package.
050B 913
050B 914 : ROUTINE VALUE:
050B 915
050B 916 : R0 = SS$_NORMAL, or called routine error status.
050B 917
050B 918 : SIDE EFFECTS:
050B 919
050B 920 : none
050B 921
050B 922 :--
050B 923
0004 050B 924 .ENTRY SHOW_A_LINE, ^M<R2>
050B 925
050B 926 ALLOC 10, R1, R2 : Allocate a descriptor & a word
051A 927 $FAOL_S CTRSTR=@4(AP), OUTLEN=8(R2), - ; Format the SHOW line
051A 928 OUTBUF=OUTDSC, PRMLST=@8(AP)
0530 929 BLBC R0, SAL_RET ; Exit if error
0533 930 MOVZWL 8(R2), R2 ; Move actual text len to descr
0537 931 MOVL OUTDSC+4, 4(R2) ; Move addr of text to descr
053F 932 PUSHL 12(AP) ; Stack spacing indicator
0542 933 PUSHAL (R2) ; ... and text descriptor
0544 934 CALLS #2, G^SCR$PUT_LINE ; Put the SHOW line to the terminal
054B 935
054B 936 BLBC R0, SAL_RET ; Exit if error
054E 937 MOVL #SS$_NORMAL, R0 ; No failing status hit
0555 938
0555 939 SAL_RET:
```

04 A2 00000000'EF 0C AC DD 0537 931
00000000'GF 02 FB 0544 934
50 00000000'8F D0 054E 937

SHODEF
V04-000

- MONITOR SHOW DEFAULT Command H 15
SHOW_A_LINE - Put a line of SHOW to term 16-SEP-1984 02:05:00 VAX/VMS Macro V04-00
5-SEP-1984 02:02:35 [MONITOR.SRC]SHODEF.MAR;1
04 0555 940 RET ; Return with status in R0
0556 941
0556 942 .END

Page 26
(20)

SUN
V04

SHODEF
Symbol table

- MONITOR SHOW DEFAULT Command

I 15

16-SEP-1984 02:05:00 VAX/VMS Macro V04-00
5-SEP-1984 02:02:35 [MONITOR.SRC]SHODEF.MAR;1Page 27
(20)

\$\$TAB	= 00000148	R	01	CDBSV_DISKAC	= 00000006		
\$\$TABEND	= 000001A8	R	01	CDBSV_DISKVN	= 00000007		
\$\$TMP	= 00000008			CDBSV_EXPLIC	= 0000000C		
\$\$TMP1	= 00000001			CDBSV_FILLER	= 0000000D		
\$\$TMP2	= 000000CF			CDBSV_HOMOG	= 00000005		
ALLCL_INT_DEFAULT	*****	X	03	CDBSV_KUNITS	= 0000000A		
ALL_STAT	= 00000000			CDBSV_PERCENT	= 00000000		
AVE_STAT	= 00000002			CDBSV_QFILLER	= 00000002		
BUILD_FAOL_ARGS	00000495	R	03	CDBSV_STD	= 00000004		
CDB	= 00000000			CDBSV_SWAPBUF	= 00000001		
CDBSA_BUFFERS	= 0000002E			CDBSV_SYSCLS	= 00000008		
CDBSA_CDX	= 00000032			CDBSV_UNIFORM	= 00000002		
CDBSA_CHDHDR	= 0000004F			CDBSV_WIDE	= 0000000B		
CDBSA_FAOCTR	= 00000004			CDBSW_BLKLEN	= 00000020		
CDBSA_ITMSTR	= 0000001C			CDBSW_DISPCTL	= 00000036		
CDBSA_POSTCOLL	= 00000026			CDBSW_QFLAGS	= 00000045		
CDBSA_PRECOLL	= 00000022			CDBSW_QFLAGS_CUR	= 00000049		
CDBSA_SUMBUF	= 0000000C			CDBSW_QFLAGS_DEF	= 00000047		
CDBSA_TITLE	= 00000010			CDBHEAD	*****	X	03
CDBSB_FAOPRELEN	= 00000041			CLASSTABLE	*****	X	03
CDBSB_FAOSEGLEN	= 00000040			CLASS_HDG	000000D3	R	01
CDBSB_ST	= 00000042			CLASS_HDR	= 00000000		
CDBSB_ST_CUR	= 00000044			CL_SEG1	000000AD	R	01
CDBSB_ST_DEF	= 00000043			CL_SEG2	000000BE	R	01
CDBSK_SIZE	= 00000053			CS_SEG1	00000027	R	01
CDBSL_BUFFERS	= 0000002A			CS_SEG2	00000040	R	01
CDBSL_ECOUNT	= 00000018			CS_SEG3	00000058	R	01
CDBSL_FAOCTR	= 00000000			CS_SEG4	00000064	R	01
CDBSL_FLAGS	= 0000004B			CS_SEG5	00000071	R	01
CDBSL_ICOUNT	= 00000014			CS_SEG6	0000007F	R	01
CDBSL_MIN	= 00000038			CS_SEG7	00000095	R	01
CDBSL_RANGE	= 0000003C			CT_STR	0000000F	R	01
CDBSL_SUMBUF	= 00000008			CURR_ERRCODE	*****	X	03
CDBSM_CPU	= 00000002			CURR_MRBPTR	*****	X	03
CDBSM_CPU COMB	= 00000008			CUR_STAT	= 00000001		
CDBSM_CTPRES	= 00000001			DEFS_A_DISP	= 0000000C		
CDBSM_DISABLE	= 00000200			DEFS_A_REC	= 00000004		
CDBSM_DISKAC	= 00000040			DEFS_A_SUMM	= 00000014		
CDBSM_DISKVN	= 00000080			DEFS_L_DISP	= 00000008		
CDBSM_EXPLIC	= 00001000			DEFS_L_REC	= 00000000		
CDBSM_HOMOG	= 00000020			DEFS_L_SUMM	= 00000010		
CDBSM_KUNITS	= 00000400			DEFS_S_DEF_DESC	= 00000018		
CDBSM_PERCENT	= 00000001			DEF_DESC	= 00000000		
CDBSM_STD	= 00000010			ERROR_QUAL	000002AF	R	01
CDBSM_SWAPBUF	= 00000002			FABSB_FNS	= 00000034		
CDBSM_SYSCLS	= 00000100			FABSC_BID	= 00000003		
CDBSM_UNIFORM	= 00000004			FABSC_BLN	= 00000050		
CDBSM_WIDE	= 00000800			FABSC_SEQ	= 00000000		
CDBSS_CDB	= 00000053			FABSC_VAR	= 00000002		
CDBSS_FILLER	= 00000013			FABSL_ALQ	= 00000010		
CDBSS_FLAGS	= 00000004			FABSL_FNA	= 0000002C		
CDBSS_QFILLER	= 0000000E			FABSL_FOP	= 00000004		
CDBSS_QFLAGS	= 00000002			FABSV_CHAN_MODE	= 00000002		
CDBSV_CPU	= 00000001			FABSV_FILE_MODE	= 00000004		
CDBSV_CPU COMB	= 00000003			FABSV_LNM_MODE	= 00000000		
CDBSV_CTPRES	= 00000000			FABSV_NAM	= 00000018		
CDBSV_DISABLE	= 00000009			FABSW_GBC	= 00000048		

SHODEF
Symbol table

- MONITOR SHOW DEFAULT Command

J 15

16-SEP-1984 02:05:00
5-SEP-1984 02:02:35VAX/VMS Macro V04-00
[MONITOR.SRC]SHODEF.MAR;1Page 28
(20)

```
FILE_HDR = 00000000
FLUSH_INT_DEFAULT ***** X 03
HOM_CLASS_PRE = 00000000
ID_STR = 0000001C R 01
IFB = 00000000
IFBSA_INPUT = 00000000
IFBSB_COL_NO = 00000004
IFBSK_SIZE = 00000005
IFBSS_IFB = 00000005
INTERVAL_DEFAULT ***** X 03
LIB$PUT_BUFFER ***** X 03
LIB$SET_BUFFER ***** X 03
MAX_CLASS_NO ***** X 03
MAX_STAT = 00000004
MIN_STAT = 00000003
MNR$_FILSYNERR ***** X 03
MNR$_SHOWERR ***** X 03
MNR_CLSSB_TYPE = 00000000
MNR_CLSSK_HSIZE = 0000000D
MNR_CLSSQ_STAMP = 00000003
MNR_CLSSS_CLASS_HDR = 0000000D
MNR_CLSSS_FILLER = 0000000F
MNR_CLSSS_FLAGS = 00000002
MNR_CLSSS_STAMP = 00000008
MNR_CLSSV_CONT = 00000000
MNR_CLSSV_FILLER = 00000001
MNR_CLSSW_FLAGS = 00000001
MNR_CLSSW_RESERVED = 0000000B
MNR_HDRSB_TYPE = 00000000
MNR_HDRSK_CLASSBITS = 00000073
MNR_HDRSK_MAXCOMLEN = 0000003C
MNR_HDRSK_REVLEVELS = 00000083
MNR_HDRSK_SIZE = 00000103
MNR_HDRSL_FLAGS = 00000001
MNR_HDRSL_INTERVAL = 00000015
MNR_HDRSL_RECCT = 00000029
MNR_HDRSO_CLASSBITS = 00000073
MNR_HDRSO_REVOCLSBITS = 00000019
MNR_HDRSQ_BEGINNING = 00000005
MNR_HDRSQ_ENDING = 0000000D
MNR_HDRSS_BEGINNING = 00000008
MNR_HDRSS_CLASSBITS = 00000010
MNR_HDRSS_COMMENT = 0000003C
MNR_HDRSS_ENDING = 00000008
MNR_HDRSS_FILE_HDR = 00000103
MNR_HDRSS_FILLER = 00000020
MNR_HDRSS_FLAGS = 00000004
MNR_HDRSS_LEVEL = 00000008
MNR_HDRSS_REVOCLSBITS = 0000001C
MNR_HDRSS_REVLEVELS = 00000080
MNR_HDRSS_TYPE = 00000008
MNR_HDRST_COMMENT = 00000035
MNR_HDRST_LEVEL = 0000002D
MNR_HDRST_REVLEVELS = 00000083
MNR_HDRSV_FILLER = 00000000
MNR_HDRSW_COMLEN = 00000071
MNR_HOMSK_PSIZE = 00000008
```

```
MNR_HOMSL_ELTCT = 00000000
MNR_HOMSL_RESERVED = 00000004
MNR_HOMSS_HOM_CLASS_PRE = 00000008
MNR_PROSB_PRI = 0000000A
MNR_PROSK_DSIZE = 0000003B
MNR_PROSK_FSIZE = 00000040
MNR_PROSK_PSIZE = 00000008
MNR_PROSK_REVODSIZE = 00000033
MNR_PROSK_REV1DSIZE = 0000003B
MNR_PROSL_BIOCNT = 0000002F
MNR_PROSL_CPUTIM = 0000002B
MNR_PROSL_DIOCNT = 00000023
MNR_PROSL_EFWM = 00000037
MNR_PROSL_EPID = 00000033
MNR_PROSL_IPID = 00000000
MNR_PROSL_PAGEFLTS = 00000027
MNR_PROSL_PCTINT = 00000004
MNR_PROSL_PCTREC = 00000000
MNR_PROSL_STS = 0000001F
MNR_PROSL_UIC = 00000004
MNR_PROSO_LNAME = 0000000B
MNR_PROSS_LNAME = 00000010
MNR_PROSS_PROCESS_CLASS = 0000003B
MNR_PROSS_PRO_CLASS_PRE = 00000008
MNR_PROSW_GPGCNT = 0000001B
MNR_PROSW_PPGCNT = 0000001D
MNR_PROSW_STATE = 00000008
MNR_SYISB_MPCPUS = 0000000D
MNR_SYISB_TYPE = 00000000
MNR_SYISK_BALSETMEM = 0000001E
MNR_SYISK_CPUTYPE = 00000026
MNR_SYISK_MPWHILIM = 00000022
MNR_SYISK_NODENAME = 0000000E
MNR_SYISK_SIZE = 0000002A
MNR_SYISL_BALSETMEM = 0000001E
MNR_SYISL_CPUTYPE = 00000026
MNR_SYISL_MPWHILIM = 00000022
MNR_SYISQ_BOOTTIME = 00000003
MNR_SYISS_BOOTTIME = 00000008
MNR_SYISS_FILLER = 0000000E
MNR_SYISS_FLAGS = 00000002
MNR_SYISS_NODENAME = 00000010
MNR_SYISS_SYS_INFO = 0000002A
MNR_SYISS_TYPE = 00000008
MNR_SYIST_NODENAME = 0000000E
MNR_SYISV_CLUSMEM = 00000000
MNR_SYISV_FILLER = 00000002
MNR_SYISV_RESERVED1 = 00000001
MNR_SYISW_FLAGS = 00000001
MNR_SYISW_MAXPRCCT = 0000000B
MON_ERR ***** X 03
MRB = 00000000
MRBSA_COMMENT = 0000002C
MRBSA_DISPLAY = 00000020
MRBSA_INPUT = 0000001C
MRBSA_RECORD = 00000024
MRBSA_SUMMARY = 00000028
```


SHODEF
Symbol table

- MONITOR SHOW DEFAULT Command

K 15

16-SEP-1984 02:05:00 VAX/VMS Macro V04-00
5-SEP-1984 02:02:35 [MONITOR.SRC]SHODEF.MAR;1Page 29
(20)

MRBSB_INP_FILES = 00000042
MRBSK_SIZE = 00000045
MRBSL_FLUSH = 00000014
MRBSL_INTERVAL = 00000010
MRBSL_VIEWING_TIME = 00000018
MRBSM_ALL_CLASS = 00000400
MRBSM_BY_NODE = 00001000
MRBSM_DISPLAY = 00000001
MRBSM_DISP_TO_FILE = 00000020
MRBSM_DIS_CL_REQ = 00000100
MRBSM_INDEFEND = 00000010
MRBSM_INP_CL_REQ = 00000040
MRBSM_MFSUM = 00000800
MRBSM_PLAYBACK = 00000008
MRBSM_PROC_REQ = 00004000
MRBSM_RECORD = 00000002
MRBSM_REC_CL_REQ = 00000080
MRBSM_SUMMARY = 00000004
MRBSM_SUM_CL_REQ = 00000200
MRBSM_SYSCLS = 00002000
MRBSQ_CLASSBITS = 00000032
MRBSQ_BEGINNING = 00000000
MRBSQ_ENDING = 00000008
MRBS\$ BEGINNING = 00000008
MRBS\$ CLASSBITS = 00000010
MRBS\$ ENDING = 00000008
MRBS\$ FLAGS = 00000002
MRBS\$ MRB = 00000045
MRBSV_ALL_CLASS = 0000000A
MRBSV_BY_NODE = 0000000C
MRBSV_DISPLAY = 00000000
MRBSV_DISP_TO_FILE = 00000005
MRBSV_DIS_CL_REQ = 00000008
MRBSV_FILTER = 0000000F
MRBSV_INDEFEND = 00000004
MRBSV_INP_CL_REQ = 00000006
MRBSV_MFSUM = 0000000B
MRBSV_PLAYBACK = 00000003
MRBSV_PROC_REQ = 0000000E
MRBSV_RECORD = 00000001
MRBSV_REC_CL_REQ = 00000007
MRBSV_SUMMARY = 00000002
MRBSV_SUM_CL_REQ = 00000009
MRBSV_SYSCLS = 0000000D
MRBSW_CLASSCT = 00000030
MRBSW_FLAGS = 00000043
NAMS\$ ESL = 0000000B
NAMS\$ ESS = 0000000A
NAMS\$ NOP = 00000008
NAMS\$ RSS = 00000002
NAMSC_BID = 00000002
NAMSC_BLN = 00000060
NAMSC_MAXRSS = 000000FF
NAMS\$ ESA = 0000000C
NAMS\$ RSA = 00000004
NAMS\$ SYNCHK = 00000003
NO_CLASS_HDG = 000000E3 R 01

OUTDSC ***** X 03
PROCDISPS = 00000005
PROCD TABLE ***** X 03
PROCESS CLASS = 00000000
PRO CLASS PRE = 00000000
QUALSA_ALC = 00000064
QUALSA_AVE = 00000074
QUALSA_BEG = 00000004
QUALSA_BY_NODE = 00000054
QUALSA_CLASS = 0000005C
QUALSA_COMM = 0000004C
QUALSA_CPU = 000000AC
QUALSA_CUR = 0000006C
QUALSA_DISP = 00000034
QUALSA_END = 0000000C
QUALSA_FLUSH = 0000001C
QUALSA_INP = 0000002C
QUALSA_INT = 00000014
QUALSA_ITEM = 000000BC
QUALSA_MAX = 00000084
QUALSA_MIN = 0000007C
QUALSA_PCEN = 000000B4
QUALSA_REC = 0000003C
QUALSA_SUMM = 00000044
QUALSA_TOPB = 0000009C
QUALSA_TOPC = 0000008C
QUALSA_TOPD = 00000094
QUALSA_TOPF = 000000A4
QUALSA_VIEW = 00000024
QUALSL_ALL = 00000060
QUALSL_AVE = 00000070
QUALSL_BEG = 00000000
QUALSL_BY_NODE = 00000050
QUALSL_CLASS = 00000058
QUALSL_COMM = 00000048
QUALSL_CPU = 000000A8
QUALSL_CUR = 00000068
QUALSL_DISP = 00000030
QUALSL_END = 00000008
QUALSL_FLUSH = 00000018
QUALSL_INP = 00000028
QUALSL_INT = 00000010
QUALSL_ITEM = 000000B8
QUALSL_MAX = 00000080
QUALSL_MIN = 00000078
QUALSL_PCEN = 000000B0
QUALSL_REC = 00000038
QUALSL_SUMM = 00000040
QUALSL_TOPB = 00000098
QUALSL_TOPC = 00000088
QUALSL_TOPD = 00000090
QUALSL_TOPF = 000000A0
QUALSL_VIEW = 00000020
QUALSS_QUALIFIER_DESC = 000000C0
QUALIFIER_DESC = 00000000
QUALPTR ***** X 03
REG_PROC = 00000000

SHODEF
Symbol table

- MONITOR SHOW DEFAULT Command

L 15

16-SEP-1984 02:05:00
5-SEP-1984 02:02:35

VAX/VMS Macro V04-00
[MONITOR.SRC]SHODEF.MAR;1

Page 30
(20)

RMS\$ SYN	*****	X	03
RV_STR	00000000	R	01
SAC_RET	00000555	R	03
SCAN_CLASSES	00000348	R	03
SCR\$PUT_LINE	*****	X	03
SCR\$SET_OUTPUT	*****	X	03
SCR\$STOP_OUTPUT	*****	X	03
SCRDSC	*****	X	03
SF_ERR	0000020B	R	03
SHC_NORM	0000048D	R	03
SHC_RET	00000494	R	03
SHD_ERR	000003EA	R	03
SHD_RET	00000402	R	03
SHODEF_CMD	00000000	RG	03
SHOW_A_LINE	0000050B	RG	03
SHOW_CLASSES	0000041E	RG	03
SHOW_COMM	000004F6	R	03
SHOW_DOUBLE	000004F4	R	03
SHOW_FAB	000000F8	R	01
SHOW_FILESPEC	000001A8	R	01
SHOW_FILE_QUAL	000002AE	R	03
SHOW_INPUT_QUAL	0000020E	R	03
SHOW_NAM	00000148	R	01
SHOW_QUAL	00000304	R	03
SHOW_SINGLE	000004F0	R	03
SHOW_SPEC_D	000002A7	R	01
SHO_END	000000FB	R	03
SHO_FILES	000001B0	R	03
SHO_FLUSH	00000178	R	03
SHO_INT	000000A2	R	03
SHO_VIEW	0000013E	R	03
SS\$ NORMAL	*****	X	03
STATS	= 00000005		
STAT TABLE	*****	X	03
SYSSFAOL	*****	GX	03
SYSSPARSE	*****	GX	03
SYS_INFO	= 00000000		
TOPB_PROC	= 00000003		
TOPC_PROC	= 00000001		
TOPD_PROC	= 00000002		
TOPF_PROC	= 00000004		
VIEWING_DEFAULT	*****	X	03

+-----+
! Psect synopsis !
+-----+

PSECT name

Allocation

PSECT No.

Attributes

. ABS .	00000000 (0.)	00 (0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE
MONDATA	000002B3 (691.)	01 (1.)	NOPIC	USR	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	NOVEC	QUAD
\$ABSS	00000000 (0.)	02 (2.)	NOPIC	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE
\$MONCODE	00000556 (1366.)	03 (3.)	NOPIC	USR	CON	REL	LCL	NOSHR	EXE	RD	NOWRT	NOVEC	BYTE

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
-----	-----	-----	-----
Initialization	29	00:00:00.09	00:00:00.58
Command processing	130	00:00:00.79	00:00:03.42
Pass 1	303	00:00:08.53	00:00:26.39
Symbol table sort	0	00:00:01.16	00:00:02.40
Pass 2	178	00:00:02.58	00:00:09.67
Symbol table output	45	00:00:00.30	00:00:01.48
Psect synopsis output	2	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	689	00:00:13.48	00:00:43.97

The working set limit was 1650 pages.

48455 bytes (95 pages) of virtual memory were used to buffer the intermediate code.

There were 50 pages of symbol table space allocated to hold 828 non-local and 40 local symbols.

942 source lines were read in Pass 1, producing 28 object records in Pass 2.

37 pages of virtual memory were used to define 22 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name	Macros defined
-----	-----
_\$255\$DUA28:[MONITOR.OBJ]MONLIB.MLB;1	4
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	0
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	14
TOTALS (all libraries)	18

1007 GETS were required to define 18 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:SHODEF/OBJ=OBJ\$:SHODEF MSRC\$:SHODEF/UPDATE=(ENH\$:SHODEF)+EXECML\$/LIB+LIB\$:MONLIB/LIB

0242 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

MONMSG
LIS

REQUEST
LIS

SHODEF
LIS

MONSUB
LIS

PREPOST
LIS

SUMMBUFF
LIS